Kurallar

Bizim ilgilendiğimiz kısım kadar anlatılması yeterlidir. Basitlik için sadece biz ve kurpiyer olan bir oyun için kuralları anlatacağım.

Bir elin değeri, eldeki kartların toplamı kadardır. Resimli kartlar (as hariç) 10 puan, sayılı kartlar ise sayısı kadar bir değere sahiptir. As kartı ise oyuncunun isteğine göre 1 veya 11 değerini alabilir. Toplam, 21’i geçtiği anda elin değeri 0 kabul edilir.

1-Hem kurpiyere hem bize (biz dediğimiz bir parti) 2’şer kart dağıtılır. Bizim kartlarımız açık, kurpiyerin 1 kartı dışında kartları kapalı durur.

2-Kart isteyip istemediğimiz sorulur. Biz kart çekmek istemeyene kadar bu döngü devam eder.

3-Artık kart çekmek istemediğimizde kurpiyer kapalı kartını açar ve toplam 17’nin altında ise 17 ve üstüne ulaşıncaya kadar kart çeker.

4-Kimin el değeri fazla ise ortadaki parayı alır ve bir sonraki tura geçilir. El değerleri eşit ise ortadaki para bölüşülür.

Bu kadar.

Beklenen değer

Beklenen değer = P1\*V1 + P2\*V2 … PN\*VN

Veya daha havalı bir gösterim olmasını istiyorsak:



s, toplam durum sayısını; n, her bir durumu; Pn n durumunun gerçekleşme olasılığını ve Vn n durumunun “değerini” belirtir. Bizim oyunumuz için bu kelimeler ne anlama gelir?

Bizim için durum çekilebilecek kart, Pn kartın çekilme olasılığı, s destedeki kart sayısı ve değer ise modifiye ettiğimiz (iyi veya kötü anlamda) kazanma olasılığımızdır.

Burada modifiye, n kartı gelirse sahip olacağımız imajiner kazanma olaslığımızdır. İmajiner dememin sebebi bu olasılığa gerçekte sahip olmamamızdır, hesaplanan sadece bir “eğer” varsayımıdır.

O zaman formülü düzenlersek:

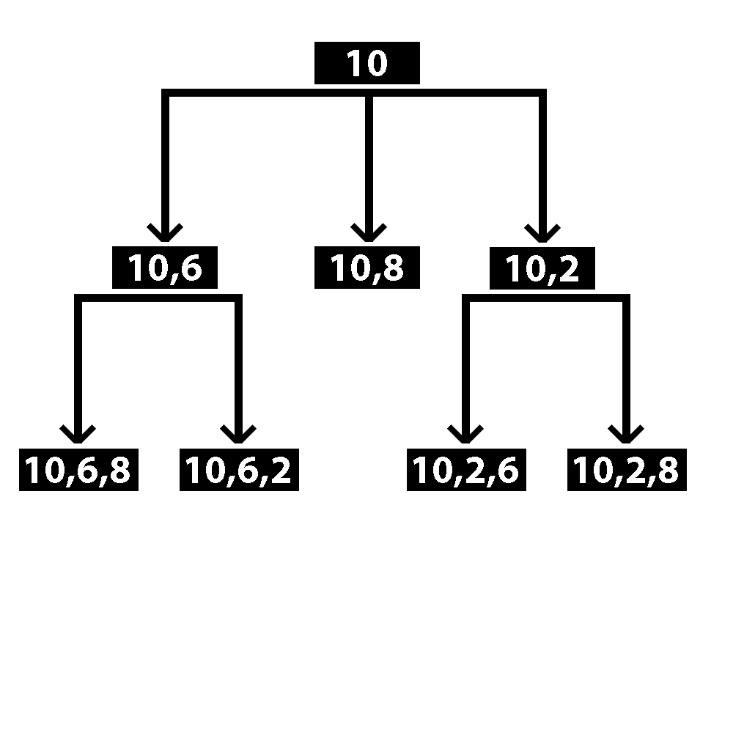
Burada Pd, düşsel kazanma olasılığını ve Pi ilk kazanma olasılığını belirtir. Eğer düşsel kazanma olasılığımız ilk kazanma olasılığımızdan düşükse bu iterasyon toplama – değer olarak yansıyacak, aksi halde ise + değer olarak yansıyacak.

Peki…İlk ve düşsel kazanma olasılığımızı nasıl hesaplarız? Ağaçlar.

Ağaçlar

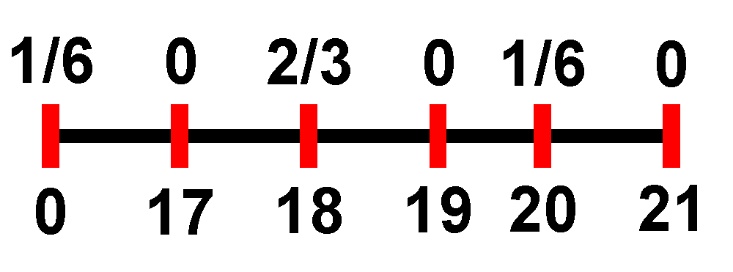
Kurpiyerin özgür iradesi ile karar vereceği herhangi bir durum olmayacağı için nihai olarak sahip olabileceği elleri hesaplamak söylemesi kolay, elde hesaplaması uzun, programlaması işkence olan bir iştir.

Örnek için oyunun sonlarına doğru destede sadece 10,6,8,2,5,4 kartlarının kaldığını düşünelim. Ardından bize 5 ve 4 kartları gelmiş olsun. Hiçbir kart çekmez isek, yani ilk kazanma olaslığımızı hesaplamak için kurpiyerin bizim bu seçimimiz sonucu sahip olabileceği elleri inceleyelim.



Destemiz bize 5 ve 4, kurpiyere de açık olarak 10 geldiğine göre bu ağacın yaprakları, kurpiyerin kurallar dahilinde sahip olabileceği nihai elleri ifade eder. (Yaprak, ağaç veri yapısında altında başka bir düğüm olmayan yani bir ebevyn olmayan düğümlere denir). Her düğümün olasılığı, ebeveyninin olasılığı \* (1/kardeş sayısı) ‘na eşittir.

Böyle bir durumda kurpiyerin 0 değerinde ele sahip olma olasılığı 1/6, 18 değerinde ele sahip olma olasılığı 2/3, 20 değerinde ele sahip olma olasılığı 1/6’dır. Daha kolay anlaşılması için bir bar oluşturmak gerek.



Üstteki sayılar olasılıkları, alttaki sayılar ise el değerlerini belirtir. Bu değerler dışında bir el değeri, kurpiyer için mümkün değildir.

Şimdi yapmamız gereken bizim el değerimizi bu bar üzerine yerleştirmek ve solunda kalan olasılık değerlerini toplamaktır.

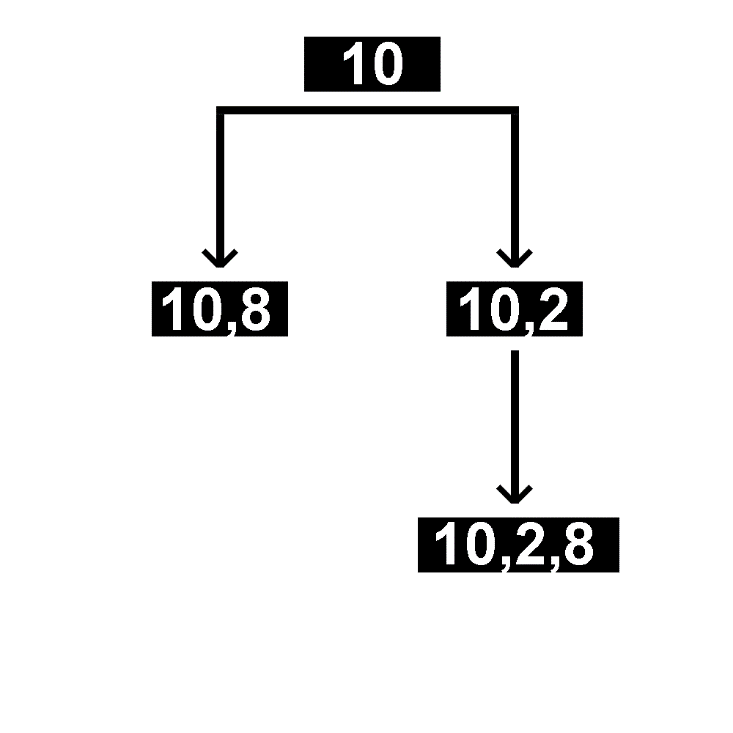
Böyle yaparsak 5+4 yani 9’u 0’ın sağına, 17’nin soluna yerleştirmemiz gerekir ve buradan kazanma olasılığımız 1/6 gelir.

Bu ne demek? Eğer hiçbir kart çekmez isek kazanma olasılığımız 1/6’dır. Bu olasılık, Pi olarak belirtilir.

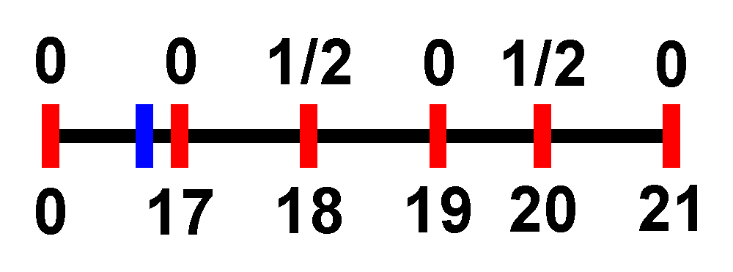
Bu işlemleri her iterasyonda tekrar yaparız. Tek fark, bir kartı imajiner desteden sileriz, ardından ağaç ve barları tekrar oluşturup çıkan imajiner kazanma olasılığına bakarız. Bunun da bir örneğini inceleyelim.

Hatırlarsak destemiz 10,6,8,2,5,4 den oluşuyordu. 5 ve 4 bize, 10 da kurpiyere (açık kart) olarak geldiğine göre kalan deste 6,8 ve 2’den oluşuyor. İlk olarak 6’yı çekeceğimiz durumda nasıl bir durum beklediğini hesaplayalım.

6’yı biz çekersek destemizin değeri 5+4+6’dan 15 olur. Kalan imajiner deste ise 8 ve 2’den oluşacağına göre ağaç şuna benzer:



Bu ağaç için olasılık/değer barımız şunun gibi olur:



Burada mavi çizgi, bizim elimizin değerini temsil eder. Görüldüğü gibi kazanma olasılığımız 0’dır.

Diğer iterasyonlar için tek tek hesaplamaları göstermeyi gerek görmüyorum.

Nihayetinde, eğer beklenen değer 0’dan büyük ise çekeceğimiz kartın kazanma olasılığımızı arttırması, küçük ise düşürmesi beklenir. Algoritmada, beklenen değer 0’dan büyük olduğu sürece Glados (Glados, bunları hesaplayıp karar veren bota koyduğum isim. Portal oyununa bir göndermedir kendisi) kart çeker.

Simülasyon ve hayal kırıklığı

Bunları c++ 17’de yazmam 1.5 ay + 2 günümü aldı. (1.5 ay, yazın blackjack’in başka bir varyasyonu için yazdığım başka bir programlama sürecini kapsıyor. Orada kullandığım fonksiyonlar, algoritma, mimari, mantık ve formülleri modifiye etmem 2 günümü aldı ve orijinal blackjack oyunu için yeterli oldu.)

Hem glados’un gerçek bir oyundan bilgilerle çalıştığı, hem de kendisine ve kurpiyere program içerisinde otomatik ve rastgele biçimde kart atandığı bir simülasyon yazdım.

Sonuçlar ise… başarısız diyemesek de beklentilerimi kesinlikle karşılamadı.

Simülasyon sonuçlarında Glados ile kurpiyer kafa kafayı gittiğini gördüm.

Bunun sebebi ne olabilir? Beklenen değer formülü eksik veya yanlış mı? Algoritması hatalı mı? Veya programımda bir sorun mu var?  
Veya bunların hiçbirisi doğru değil ve bu oyun için kumarhanenin avantajı bunlara rağmen çok mu fazla?  
  
blackjack\_real programı için <https://github.com/Sokisati>

Not: Şu ana kadar (kişisel takıntıdan ötürü) yazdığım c++ programlarının her algoritmasını, mimarisini ve kodunu kendim geliştirdim. Bu, çok doğru bir tutum olmasa da öğrenme aşamasında bana fayda sağlayacağını düşündüğüm bir kural. Defalarca kez fonksiyon çıktılarını konsola yazdırıp kontrol ettim ve hataların hepsini çözdüğümü düşünüyorum ama gene de programda bug veya glitch’ler pekala var olabilir.